An Investigation of Target Acquisition with Visually Expanding Targets in Constant Motor-space

November 15, 2005

Philip J Brock
Dr Andy Cockburn (supervisor)

Abstract

Target acquisition is a core part of modern computer use. Fitts' law has frequently been proven to predict performance of target acquisition tasks; even with targets that change size as the cursor approaches. Research into expanding targets has focussed on targets that expand in both visual- and motor-space. We investigate whether a visual expansion with no change in motor-space offers any performance benefit. We investigate constant motor-space visual expansion in both abstract pointing tasks (based on the ISO9241–9 standard) and in a realistic deployment of the technique within fisheye menus. Our fisheye menu system eliminates the 'hunting effect' of target acquisition observed in Bederson's initial proposal of fisheye menus, and in an evaluation we show that it allows faster selection times and is subjectively preferred to Bederson's menus. We also show that visually expanding targets can improve selection times in target acquisition tasks, particularly with small targets.

Contents

1 Introduction **1**

2 Related Work **3**
2.1 Target Acquisition . . . 3
2.1.1 Expanding Targets . . . 4
2.2 Fisheye Effects . . . 4
2.3 Menus . . . 5
2.3.1 Fisheye Menus . . . 7

3 System Design **11**
3.1 Fisheye Menus . . . 11
3.1.1 Occluding Fisheye Menus . . . 11
3.1.2 Rearranging Fisheye Menus . . . 12
3.2 Menu Testing Interface . . . 13
3.3 Abstract Targets Testing Interface . . . 14

4 Controlled Experiments **17**
4.1 Experiment One—Menu Evaluation . . . 17
4.1.1 Method . . . 17
4.1.2 Procedure . . . 18
4.1.3 Results . . . 19
4.1.4 Discussion . . . 20
4.2 Experiment Two—Abstract Targets . . . 20
4.2.1 Method . . . 21
4.2.2 Procedure . . . 21
4.2.3 Results . . . 22

5 Discussion and Further Work **27**
5.1 Further Work . . . 27
5.1.1 Abstract Targets – Experimental Design . . . 27
5.1.2 Fisheye Menus – Cueing Systems . . . 28
5.1.3 Fisheye Menus – Cascading Menus . . . 28
5.1.4 Fisheye Menus – Long Menus . . . 28
5.1.5 Fisheye Menus – the Vision Impaired . . . 28
5.1.6 Fisheye Menus – Longitudinal Study . . . 28

6 Conclusions **31**
6.1 ACKNOWLEDGMENTS . . . 31

Chapter 1

Introduction

Target acquisition is an essential part of everyday computer use, and little meaningful interaction is possible with a pointing device if users are not adept at this task. As display technology has developed, screen resolutions have increased, allowing developers to provide complex and highly detailed environments. With this has come an increase in the number of small user interface widgets such as clickable buttons, pull-down menus, window borders and others. Unfortunately for users, acquiring these small targets can be difficult (Cockburn & Firth 2003). In a WIMP (Windows, Icons, Menus and Pointing device) environment, such target acquisition tasks are carried out frequently, so even a small improvement in acquisition time will have a large effect on overall user productivity.

Fitts' law (Fitts 1954) is frequently used to evaluate user performance in selecting targets (Soukoreff & MacKenzie 2004) and can be used to predict the time a user will take to move the mouse cursor from a starting point to a goal. Fitts' law assigns an index of difficulty to each targeting task based on the distance and size of the target—the smaller and more distant it is, the greater the index of difficulty. It has been consistently shown that there is a close correspondence between the index of difficulty and the movement time taken to select the target (Fitts 1954, Soukoreff & MacKenzie 2004, Zhai, Conversy, Beaudouin-Lafon & Guiard 2003, McGuffin & Balakrishnan 2002, Cockburn & Firth 2003, Blanch, Guiard & Beaudouin-Lafon 2004*a*, MacKenzie 1992). Any evaluation undertaken will necessarily include a Fitts' analysis.

A promising strategy for aiding target acquisition is the use of expanding targets—as the cursor approaches a target it expands to facilitate the user clicking on it. Studies by McGuffin & Balakrishnan and Zhai et al. have shown that expanding targets have a positive effect on target acquisition performance. Both of these studies used expanding targets that underwent a corresponding expansion in motor-space, but an alternative strategy may be to implement system using visual expansion only to aid acquisition.

In an application of visually expanding targets Bederson implemented a 'fisheye menu' system. A fisheye menu works by providing a dynamic magnification effect of the menu items closest to the mouse cursor. There is an additional effect, that as the number of menu items increases, the unmagnified size of each item decreases, thus allowing for more menu items in the same screen space. The potential benefits are faster acquisition times because of target expansion and longer displayable menus. The evaluation showed that fisheye menus have potential, but failed to deliver empirical evidence of their effect on user performance. Furthermore, the implementation suffered from a common problem with fisheye effects (Blanch et al. 2004*a*), whereby the fisheye lens causes the surrounding targets to move around, thus making target acquisition more difficult.

We are investigating the use of visually expanding targets with no increase in motor-space, and an alternative implementation of fisheye menus that uses neighbouring item occlusion when expanding, rather than menu rearrangement as was done in the original fisheye menus (Bederson 2000).

In Chapter 2, we describe related work on target acquisition, in particular that involving expanding targets, the application of Fitts' Law to target acquisition tasks, and applied visual expansion in the form of fisheye menus. Following this is a description of an alternative fisheye menu we have developed, and an overview of the evaluation interfaces. We then present our empirical investigation of visually expanding targets, and an evaluation of menu selection comparing two different fisheye menus and a traditional static view menu.

Chapter 2

Related Work

2.1 Target Acquisition

Target acquisition is an ever present task for computer users, and there has been extensive research into how interfaces can be designed to help users perform targeting tasks faster.

Fitts' law (Fitts 1954) is frequently used to evaluate user performance in selecting targets (Soukoreff & MacKenzie 2004) and can be used to predict the time a user will take to move the mouse cursor from a starting point to a goal. Fitts' law assigns an index of difficulty to each targeting task based on the distance and size of the target—the smaller and more distant it is, the greater the index of difficulty. It has been consistently shown that there is a close correspondence between the index of difficulty and the movement time taken to select the target (Fitts 1954, Soukoreff & MacKenzie 2004, Zhai et al. 2003, McGuffin & Balakrishnan 2002, Cockburn & Firth 2003, Blanch et al. 2004*a*, MacKenzie 1992).

There are a number of different formulations of Fitts' law. For the purpose of Fitts' law analysis, we will use the Shannon formulation of Fitts' law proposed by Soukoreff & MacKenzie (2004) as a standard for HCI researchers. This formulation models movement time (MT) as:

$$MT = a + blog_2(\frac{A}{W} + 1) \tag{2.1}$$

Where A is the amplitude (distance) and W the width of the target. The advantages of the Shannon formulation are that it cannot produce negative values of ID, it provides a better fit with observed data, and it is less likely to result in negative intercepts that indicate negative selection times for low IDs (MacKenzie 1992).

Soukoreff & MacKenzie (2004) makes a set of recommendations to HCI researchers conducting Fitts' law analyses. Their goals are to improve the robustness of Fitts' models and improve the comparability and consistency of publications. We will follow these as closely as possible in the Fitts' analysis presented in this paper.

A detailed discussion of Fitts' law is beyond the scope of this paper, however Soukoreff & MacKenzie (2004) and Cockburn & Brewster (2005) present a good overview of the current state of Fitts' law usage.

Various techniques have been used to help improve target acquisition. Cockburn & Brewster (2005) investigated small target acquisition assisted by multimodal feedback, evaluating all combinations of non-speech audio, tactile and pseudo-haptic 'sticky' feedback. Tactile feedback was supplied by vibration of the mouse, and the sticky feedback by adjusting the mouse control-display gain when the cursor entered the target. For an abstract targeting task their results showed that all kinds of feedback result in reduced acquisition times, but when applied to a menu selection task some combinations of feedback did not perform well, and overwhelmed the user with excessive feedback that interfered with target acquisition.

Another approach to enhancing target acquisition is through modified cursor behaviour. Object pointing uses a screen cursor that skips empty spaces, reducing the amount of information needed to be conveyed to the system by the user (Guiard, Blanch & Beaudouin-Lafon 2004). The cursor operation is thereby changed from an act of positioning a pointer and establishing which GUI element lies beneath it, to simply choosing from among the available GUI elements. The bubble cursor (Grossman & Balakrishnan 2005) is an area cursor that dynamically resizes its activation area in such a way that it always encloses one target (and no more). Although evaluations of bubble cursors have proven them efficient for selecting abstract targets, there are, as yet, no known real-world applications.

Semantic pointing (Blanch et al. 2004*a*) uses a modification of the mouse control-display gain to effectively change the size of targets in motor-space, similarly to the sticky targets used in Cockburn & Brewster (2005). In an evaluation of an interface using semantic pointing Blanch et al. find that Fitts' law models the target acquisition times based on the effective motor-space size of the targets rather than their visual representation. Some guidelines are proposed, suggesting the use of semantic pointing to guide user behaviour, such as increasing the size of the "Save" button when displaying an "Unsaved changes" warning dialogue box.

Cockburn & Firth (2003) evaluated three different methods to aid the acquisition of small targets. One of the methods used was that of 'bubble' targets, which increase the width of the target as the cursor approaches. It was found that bubble targets resulted in improved user performance compared to traditional statically sized targets[1].

2.1.1 Expanding Targets

The use of expanding targets to improve target acquisition has been the subject of several research projects. An empirical study by McGuffin & Balakrishnan (2002) showed that expanding targets improve acquisition times, even when the target starts expanding when the cursor is 90% of the way through its movement towards the target. A follow up study by Zhai et al. (2003) repeated the evaluation but included a random block where targets could expand, remain the same size or shrink as the cursor approached, and showed that expanding targets aid targeting even in the absence of expectation.

Despite showing considerable promise, there have been few applications of expanding targets within GUIs (Blanch, Guiard & Beaudouin-Lafon 2004*b*). This is largely due to the fact that increasing the size of a target causes the need for other targets to shrink, resulting in the moving targets described in the following section on fisheye effects. A key advantage of visually expanding targets is that when used in conjunction with occlusion no rearranging of GUI elements is necessary.

2.2 Fisheye Effects

Fisheye views provide a focus+context view of an information space, providing "a balance of local detail and global context" (Furnas 1986). Information is filtered through the use a degree of interest (DOI) function that assigns a value to each item of information based on how interested the user is in seeing the information, given the current task. Only those items meeting the threshold value are displayed in the view. Furnas implemented a code browser using a simple fisheye view to convey structural information to the programmer. The DOI functions outlined by Furnas are based upon the distance between one item and another, assigning higher value to closer items, however other DOI functions need not be constrained to work in this way, and fisheye views can provide a very flexible view of data. There have been numerous applications of the fisheye technique, and they have been used in word processors, menus, calendars, software visualisations, maps and document browsers among others (Gutwin 2002, Bederson 2000).

Figure 2.1: The MacOS X dock uses a fisheye effect to expand items nearest to the cursor

[1]The authors had some reservations about this widget type and commented: "Due to the risks of visual distraction and of encouraging hunting, we recommend that expanding targets be used with caution."

Many fisheye effects use visual expansion of a focus region to provide detail. An example of this is the MacOS X 'dock' icon panel, shown in Figure 2.1, which uses a fisheye distortion to assist users in targeting items. As the cursor approaches the dock, the icons nearest to the cursor expand in size, pushing neighbouring icons outwards. If the cursor is run along the dock from end to end this rearranging effect can be clearly seen, and results in considerable sideways movement of the icons. A consequence of this is that users receive no performance benefit from the expansion, as any improvement offered by the increased target size is negated by the effort of having to select moving targets (McGuffin & Balakrishnan 2002, Zhai et al. 2003). The sideways movement of neighbouring icons matters little if the cursor approaches the target item directly on a vector perpendicular to that of the dock, but in all other cases it has a negative effect. An alternative method of expansion is recommended by both McGuffin & Balakrishnan (2002) and Zhai et al. (2003), whereby the icons undergo an in place expansion, occluding neighbouring icons if necessary, and providing the benefits of the fisheye without the drawbacks of the rearrangement.

A common problem with magnified fisheye views is difficulty in focus-targeting, where a user-controlled focus point is moved to a new location (Gutwin 2002). This is experienced as target objects moving as the focus approaches them, making them significantly harder to acquire. To make matters worse, this effect is experienced most in the focus region, which is precisely the area the user is most interested in. Gutwin (2002) presents speed-coupled flattening as a way to deal with this problem. Speed-coupled flattening takes the velocity and acceleration of the user's pointer into account when determining the distortion level of the fisheye view, reducing the distortion effect when the pointer is moving rapidly, and restoring the distortion level when the pointer slows again. In many ways this is similar to speed dependent automatic zooming (SDAZ), a method for scrolling documents that automatically changes the level of zoom in accordance with the scroll speed (Cockburn, Savage & Wallace 2005). At slow speeds the document remains at full zoom, as the scroll speed increases the view zooms out, maintaining the view and providing the context of the scrolling operation. Gutwin evaluated the use of speed-coupled flattening, and found that it resulted in reductions in targeting time and error-rates.

2.3 Menus

Menu systems are prevalent throughout many graphical user interfaces and there has been substantial research into how they are used and how they can be improved (Byrne, Anderson, Douglass & Matessa 1999, Ahlstrom 2005, Nilsen & Evans 1999, Findlater & McGrenere 2004, Callahan, Hopkins, Weiser & Shneiderman 1988). This is well-motivated, as selecting menu items is a frequent and potentially time consuming task.

There are a number of different well-established HCI laws that can provide a good indication of what to expect from tasks involving selecting items from drop-down menus. The Hick-Hyman law for choice reaction (Hick 1952, Hyman 1953) describes the time taken to make a selection decision as a function of the number of available choices. The reaction time (RT) is given by Equation 2.2:

$$RT = a + blog_2(n+1) \tag{2.2}$$

Where n is the number of equiprobable choices, and a and b are empirically determined constants. The $+1$ component represents the uncertainty of whether or not to respond—essentially, whether the task has started or not, and thus the person performing the selection is effectively grappling with one more option than is presented. Hick-Hyman law has been successfully applied to selecting items from menus (Landauer & Nachbar 1985), allowing us to infer that selections from longer menus will require more time on average than those from shorter menus of the same type.

As discussed previously, Fitts' law predicts that smaller, more distant targets will take longer to acquire. So, given the equally sized items in a standard static pull-down menu we can predict that the movement time required to select the menu item in the tenth position will be greater than that required to select the first menu item. Thus, given Hick-Hyman and Fitts' law we can predict that in general, the time to recognise a target menu item will be greater when using longer menus, and the time taken to select it will increase as the item's position in the menu increases.

Steering law (Accot & Zhai 1997) can be used to model trajectory-based interactions, where a user must navigate within the confines of a set path. This situation arises with cascading pull-down menus, as users follow a path from the top of the menu to their target item, and often having to change from vertical to horizontal movements of the cursor (Kobayashi & Igarashi 2003). The evaluation presented in this paper uses menus with only one level of menu items (there are no cascading submenus), so a steering law analysis is not required.

Considerable research has been done with the goal of finding the optimal ordering of menu items within pull-down menus (Sears & Shneiderman 1994, Card 1982, Somberg 1987). In static pull-down menus the ordering of menu items can be decided in a number of ways. Menus may be ordered alphabetically, numerically, logically, functionally, by frequency of selection or even randomly (Sears & Shneiderman 1994). Somberg (1987) found that when users are unfamiliar with a menu a categorical (functional) ordering is most efficient, followed by an alphabetical ordering. Both Somberg (1987) and Card (1982) found that random orderings were least efficient, but that with repeated exposure to the menus ordering ceased to matter significantly as participants learned the spatial location of menu items.

One of the possible orderings of menu items is to order them by frequency of selection, with the most frequently used menu item at the top of the menu and the least frequently used at the bottom. Mitchell & Shneiderman (1989) investigates the performance of these dyanamic menus with traditional static menus, finding that intially users are slower with dynamic menus, but adjust to them so that later there is no difference in speed between them. However, subjectively the vast majority of participants favoured using static menus, with 81% preferring them to dynamic menus. Mitchell & Shneiderman note that “Some subjects were disoriented and experienced strong negative affect when the menu ordering changed”, and suggest that where adaptive systems are used, users should be able to exercise some control over the adaptive behaviour.

Split menus are a variation of standard drop-down menus that reserve a portion at the top of the menu for the most frequently selected items. In an evaluation, Sears & Shneiderman (1994) found that selection times were 17 to 58% faster using split menus compared to alphabetically ordered drop-down menus, and that split menus were subjectively preferred by participants. In a follow up study Findlater & McGrenere (2004) compared static split menus with adaptive and adaptable split menus. In the static condition the items in the top partition were those occurring most frequently in the experimental tasks, in the adaptable participants were responsible for customising which items appeared in the top partition, and in the adaptive the items are changed dynamically based on frequency of selection up to that point. They found that static split menus are faster than adaptable, which are in turn faster than adaptive split menus, but that subjectively participants preferred controlling the customisation themselves.

Alternatives to drop-down menus have also been developed. Pie menus place menu items in a circular arrangement at equal radial distances from the centre (Callahan et al. 1988), thereby reducing selection times as only a minimal movement is required to select an item. Callahan et al. (1988) evaluated the performance of pie menus against drop-down menus and that for menus containing 8 items, pie menus allowed for faster selection and lower error-rates than drop-down menus. Although pie menus can be faster, they are constrained by only being able to display a limited number of menu items around the circle, and do not scale well to larger menus (Moyle & Cockburn 2003).

A modified version of pie menus known as marking menus has also been developed. These allow users to select menu items either by displaying the pie menu, or by simply making a straight mark in the direction of the target menu item without displaying the menu at all (Kurtenbach & Buxton 1994). In an experimental evaluation, Kurtenbach & Buxton (1994) found that marking menus are used first as pie menus, with users moving towards using marks as they gain expertise and knowledge of the menu layout. Once learned, using marks was 3.5 times faster on average than selecting using the displayed menu. Although a promising menu system, Kurtenbach & Buxton recommend that it only be used for menus that are unlikely to change, as users would be unable to use marks effectively if the menu layout cannot be memorised.

Work has also been done to improve cascading menus. Kobayashi & Igarashi (2003) implemented a cascading menu system with a modified direction-based submenu pop-up behaviour. Typically to open a submenu users must move the cursor to the far right (or left) side of the menu, where the pop-up submenu is then displayed. Selecting some menu items means performing a series of vertical and horizontal movements to navigate the menu hierarchy, which can be a complex steering task (Accot & Zhai 1997). In the direction-based system, only reduced horizontal movements are required, as this signals the menu system to immediately display the pop-up menu at the location of the horizontal movement. Direction-based submenu pop-ups were found to allow 12% faster selection times on average, and significantly reduced the length of the movement path for each selection. An alternative strategy proposed by Ahlstrom is the use of "force fields", a warping effect that attracts the mouse cursor towards submenus, making it easier to select them and decreasing selection times by 18% on average (Ahlstrom 2005).

2.3.1 Fisheye Menus

Fisheye menus were first proposed by Bederson (2000) as a way of supporting long menus without the need for buttons, scrollbars or hierarchies. As with other fisheye-distortion interfaces, fisheye menus provide an enlarged view of a focus area, whilst still displaying the context in which it occurs, with a transition area in between. In effect this means that those menu items closest to the mouse cursor are drawn in a large, easily readable font, while those more distant are drawn in a reduced font size. This allows for menus with many more items than are possible with static pull-down menus, at the cost of diminished readability of those menu items not in the focus area.

The fisheye menus were implemented to use as much as possible of the available screen space. This means that if all of the menu items can be drawn at full size, they are, and there is no fisheye effect. The focus length of each menu can be set, and this controls how many items are drawn at full size around the cursor. In the case where this would use more space than is available to the menu the focus length is reduced, and if this adjustment is insufficient, the maximum font size is also decreased. In all cases the inter-item spacing is largest in the focus area—those items shown at the minimum font size have very little inter-item spacing. Fisheye menus are essentially densely-packed visually expanding targets, as the motor-space mapping is done based on the minimum size of the menu items—any expansion that occurs is only visual and does not include a corresponding increase in clickable area.

The Hunting Effect

The fisheye effect of fisheye menus is achieved by rearranging the layout of the menu as the cursor is moved over it. The focus area centres on the cursor's current location, and the menu items' sizes dynamically change to accommodate the moving focus area. As the cursor moves down the menu, the menu items at the top become smaller as the lower menu items increase in size. This results in the amount of space used to display the menu staying constant, with only the amount allocated to each menu item needing to change.

An unfortunate side effect of this rearranging strategy is a "hunting effect" caused by users trying to select moving targets. As previously mentioned, this problem is common in fisheye views (Gutwin 2002) and occurs here because menu items seem to move in the opposite direction to movement of the mouse cursor—coming towards the mouse cursor as it moves towards them, and receding from it as moves away. The effect is clearly visible in Figure 2.2, which shows the consequence of the hunting effect as a user attempts to select a menu item in a fisheye menu. As the user moves the mouse towards the target menu item (My Help Desk) the menu spacing and sizing is being rearranged, so that when they arrive at the original position of the target it is no longer there. The nett result of this is that users are seriously hampered when attempting to rapidly target menu items, particularly with the ballistic movements that occur when acquiring targets that are nearby (Gann & Hoffmann 1988).

Bederson noticed this effect and implemented two different strategies to mitigate it. The first is that fisheye menus employ an alphabetic index displayed on the left side of the menu. The positions of the alphabetic indexes are calculated to reflect the position of the first menu item beginning with that letter were the item to be centred in the focus area. Because the menus are alphabetically ordered users can quickly position the cursor near the target menu item based on the position of its initial letter in the index, without ever having to read the individual menu items. As a direct result of this, fisheye menus require menu items to be sorted alphabetically before being added to the menu, reducing the number of possible applications of fisheye menus.

The second strategy for reducing the hunting effect is the focus lock mode. To use this mode, users move the cursor until it is close to a target menu item. They then move the cursor to the right side of the menu, and enter the focus lock mode. In this mode the rearranging behaviour of the menu is temporarily disabled and the items in the focus are locked, allowing the user to accurately select the target item without it moving. The focus area, which is indicated by a dark grey rectangle, can be extended by moving the cursor above or below the current focus region, and items will be added to focus area at their maximum size. Moving the cursor back into the left side of the menu removes the focus lock and the menu returns to normal behaviour. The fisheye menus' focus lock mode adds extra complexity to menu usage and can frustrate users' targeting, as illustrated in Figure 2.3, which shows how a user-slip while using the focus lock can cause the target menu item to be lost.

In a limited evaluation involving ten participants, three untimed selection tasks, an open-ended browsing task, and a questionnaire, Bederson compared fisheye menus to hierarchical, scrolling arrowbars and scrollbar menus. The results showed that subjectively, participants preferred fisheye menus above all others for browsing tasks, and ranked it a close second to hierarchical menus for goal-directed tasks. Unfortunately, the evaluation stopped short of recording selection times or error-rates, and so little is known about how well fisheye menus actually compare to other menu types.

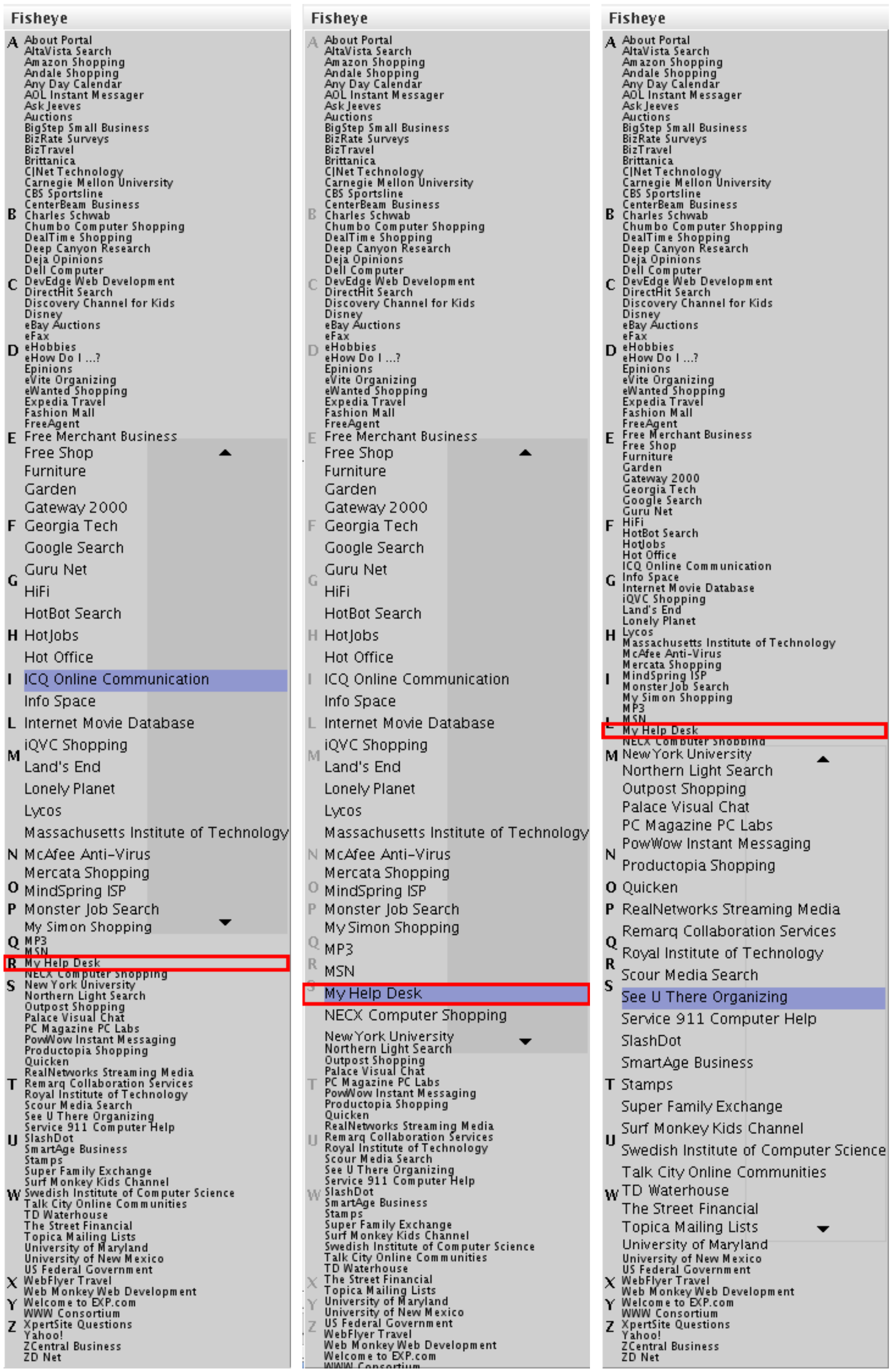
The rest of this paper describes an evaluation of visually expanding targets, and the implementation and evaluation of an alternative fisheye menu system. Our aim is to develop a fisheye menu using in place expansion of menu items, with the larger items occluding those smaller than them if necessary. The menu is identified as distinct from Bederson's fisheye menus with the name "occluding fisheye menus", and avoids the hunting effect present in Bederson's implementation, while still maintaining the positive aspects of fisheye menus.

(a) The menu is first posted

(b) Overshooting—the cursor is moved to where the target used to be

(c) Correction—the cursor is moved back and over the target menu item

Figure 2.2: Fisheye menus showing a hunting effect. The target item is indicated with a boxed outline. As the cursor approaches the target, the menu is rearranged and the target moves upwards—forcing the user to “hunt” for the moving target.



(a) Entering focus lock mode (b) Over the target menu item (c) Leaving focus lock mode

Figure 2.3: The fisheye menus' focus lock can frustrate targeting. The target item is indicated with a boxed outline. The focus lock mode is entered and the cursor moved to the targeted item, accidentally leaving the focus lock mode causes the menu to be redisplayed.

Chapter 3

System Design

This chapter explains the design and implementation of the occluding fisheye menu widget, and the evaluation interfaces used for conducting the experiments.

3.1 Fisheye Menus

Fisheye menus are an example of densely-packed visually expanding targets, and are an ideal candidate for an ecological evaluation to see if visually expanding targets have practical applications.

Bederson (2000) was the first to implement fisheye menus, and did so using the Java language and API. Because of the need to do a comparison between Bederson's implementation and that proposed by this paper, the occluding fisheye menu widget was also implemented using Java.

Some slight modifications had to be made to Bederson's fisheye menus, as detailed later in this section. To disambiguate, the modified fisheye menus are called "rearranging" fisheye menus, and the menus we have created are "occluding" fisheye menus.

3.1.1 Occluding Fisheye Menus

Occluding fisheye menus are intended to offer the benefits of fisheye menus without the drawbacks. There are a number of key differences between occluding fisheye menus and the original fisheye menus that are worth noting. The primary change is that the visual expansion of targets now occurs in place, rather than by rearrangement. It is by this that occluding fisheye menus get their name, as rather than rearrange the menu, expanded items will occlude their neighbours if necessary. The immediate impact of this is that occluding fisheye menus do not suffer the hunting effect that the original fisheye menus do. As can be seen in Figure 3.1, the target menu item remains in the same place even when it is expanded, making it far easier to acquire.

The second major change is that the focus lock has been removed. With menu items no longer changing position it no longer makes sense to have the focus lock, which would only add unneccessary complexity to the menu system. As described in Chapter 2, the focus lock can confound user targeting every bit as much as the hunting effect.

The final major change is that the alphabetic indices have been removed. Occluding fisheye menus are intended to function in a variety of situations, not just the list of data items suggested as an application for the original fisheye menus (Bederson 2000). Given the wide range of possible menu orderings it is best to postpone this decision to the implementor of a system, rather than imposing a specific ordering requirement on developers.

The occluding fisheye menu uses the same amount of vertical space as an ordinary JMenu, as long as space is available to use. When the number of menu items goes beyond what will fit in the available space the minimum font size is decreased. This is the font size that unexpanded menu items are drawn in, and also represents the size of the menu items in motor-space, as the vertical space is divided equally between the menu items whether expanded or not. When the number of menu items is within the bounds of what

(a) The menu is first posted

(b) The target is selected—the cursor is moved to the target, which is expanded in place

Figure 3.1: Occluding fisheye menus avoid the hunting effect. The target item is indicated with a boxed outline. As the cursor approaches the target, the menu items undergo in place expansion, allowing the user to select them based on their original position.

an ordinary JMenu could display, the occluding fisheye menus display as a JMenu would, only with the addition of the expanded items around the focus area.

Standard menu operations are supported, such as clicking and dragging within the menu, and as an extended JMenu, occluding fisheye menus can be used anywhere a JMenu is currently.

3.1.2 Rearranging Fisheye Menus

Although we wanted to compare our newly designed occluding fisheye to Bederson's original fisheye menus, a number of modifications were required, both to make the menus comparable and to ensure they actually behaved in a well-defined and error-free manner. We call the modified version of Bederson's menus rearranging fisheye menus, to distinguish them from the behaviourally different occluding fisheye menus.

The first modification was to remove the alphabetic indices on the left side of Bederson's menus. We are evaluating fisheye menus as a general purpose menu system, and as discussed in Chapter 2, the ordering of menu items within a menu can be done in a number of ways—enforcing an alphabetic ordering unneccessarily restricts the possible uses of the menus.

The default behaviour of Bederson's menus is use as much space as is available for menus, up to the point where every menu item is displayed at the maximum (expanded) font size. To make the menus comparable the function that calculates the amount of vertical space to use was modified to report the amount of space a JMenu containing the menu items would use. The layout manager for the menus then created the desired fisheye effect.

A number of changes had to be made to the menus to bring them in line with current Java 1.5 behaviour. This included changing the container the menus are displayed in from a JWindow to a JPanel, and changing the colours used to reflect Java Swing's current "look and feel". The method of calculating the width of the menu was flawed and produced menus that could not always display the longer menu items, this was corrected. The width of the highlight indicating the currently selected menu item was extended to cover the whole menu. Finally, we ensured that standard JMenu behaviour was supported, such as being able to click and drag within the menu, and correct menu unposting behaviour.

The focus lock and rearranging behaviour of the menu was left intact, complete with the hunting effect outlined in Chapter 2.

3.2 Menu Testing Interface

The menu testing interface evaluated participants' performance using three different menu types. The three menus presented were the occluding fisheye menu proposed by this paper, the rearranging fisheye menus described above, and a standard static-view menu implemented with Java Swing's JMenu widget.

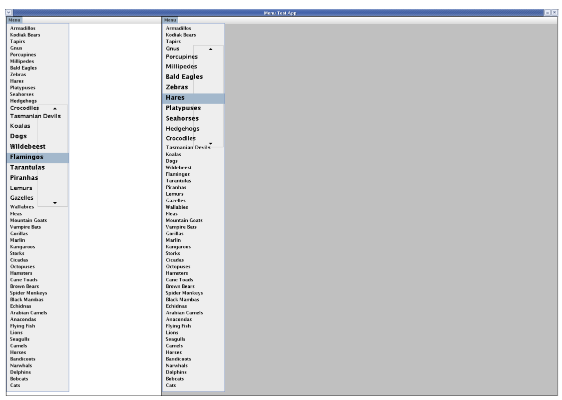


Figure 3.2: The menu evaluation interface, showing rearranging fisheye menus. On the left side is the cueing menu, and on the right the active target menu.

The testing interface was divided into two sides, with the cueing region on the left and the active target menu on the right. The cueing region displayed two kinds of cues, a menu cue, which showed the target menu item as it would appear when selected in the active menu, and a word cue, which showed the name of the target menu item to be found in the active menu. The testing interface is shown in Figure 3.2, displaying a menu cue for a rearranging fisheye menu containing 50 menu items. Figure 3.3 shows the testing interface displaying a word for an occluding fisheye menu.

The operation of the interface is simple, the cue is displayed and the participant selects the matching menu item from the target menu. If the selection was correct then the interface progresses to the next task, if the selection was wrong then an error is recorded, the background region of the target area turns red, and the participant must select the correct menu item as quickly as possible. The interface records the time of the successful selection, and the length of time the cursor was over the target item before it was selected. The participant's name, selection time, task number, menu length and menu type are all logged for later analysis.

Tasks are separated by cue type, so all tasks with the menu cue are completed before starting tasks with the word cue. Within each cue type the order of presentation of the menu types is randomised, and all tasks are completed with that menu type before progressing to the next. Tasks are completed for different menu lengths with each of the cue types. The menus are populated from a list of words read in from a text file when the testing interface is first started. The testing interface ensures that the same word does not occur twice within a given menu. When using the menu cue the menu items remain the same for all tasks with the same menu length. When using the word cue the menu items are randomly generated for each and every

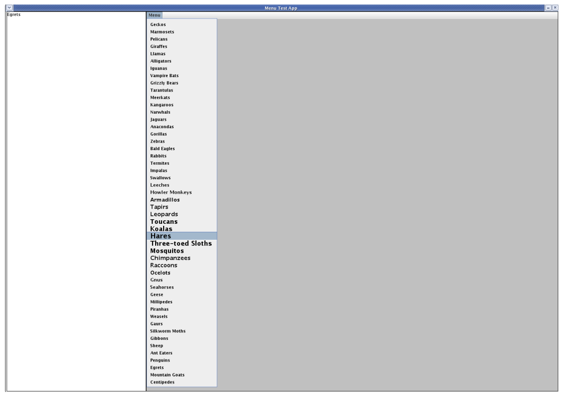


Figure 3.3: The menu evaluation interface, showing occluding fisheye menus. On the left side is the word cue, and on the right the active target menu.

task, creating a condition where the user has never encountered the menu before. The menu lengths to use, and tasks to perform are configured before the evaluation according to experimental needs.

3.3 Abstract Targets Testing Interface

The tasks used in Fitts' law analyses frequently use the Fitts' paradigm, a one-dimensional, horizontal targeting task, where selections of vertical bars are timed. The ISO9241–9 standard, however, advocates a multidirectional targeting task, using a circular array of targets to evaluate pointing interfaces. This ensures that selections are made from numerous different directions, as happens with the components of deployed GUIs, and reduces the chances that an interface that favours selections from a certain direction will go unnoticed. For this reason the abstract targets testing interface was built to use circular arrays of targets. The path participants follow starts and ends at the topmost target in the circle, and they select alternating targets clockwise around the circle. Each target, once selected, becomes the starting point for the next task, which is selecting the target on the opposite side of the circle.

The interface supports targets with multiple types of feedback. For the purposes of the experiment, which is outlined in more detail in the following chapter, four different target types were required. These were visually expanding, visually expanding and colour changing, static, and static and colour changing. Due to the fact that the visually expanding targets undergo no expansion in motor-space it was decided to delay the expansion until the cursor was above the target, which is different from the expansion policy employed in other expanding targets evaluations (Zhai et al. 2003). Likewise, the expanded target must contract as soon as the cursor leaves the clickable area of the target. Without this, participants could erroneously click on a part of the target that was outside of the clickable area, which was deemed to be unacceptable and useless in practical terms. The colour changing feedback is also limited to those times when the mouse cursor is over the target, making the two types of feedback comparable. This is also the standard behaviour of many button panels in use in GUIs, which highlight the button currently under the cursor.

The interface creates blocks of tasks, which can consist of one or more circular arrays. The number of targets to display around the circle, the types of targets to display, the size of the targets and the radius of the circle are all highly configurable, so the interface can test a wide range of conditions. Three different base configurations are supported: the first allows for target arrays all of the same target type, the second for arrays with randomly selected target types, and the third for arrays with balanced, random-ordered permutations of target types. Essentially this means four arrays of targets are created for each condition, with the target types randomly distributed throughout each of the four arrays. An additional constraint we imposed was that an equal number of each target type must occur in each wheel, and every possible array position be used once and only once by each target type.

When the interface is run it displays a dialogue asking for the user's name. The log file is then identified

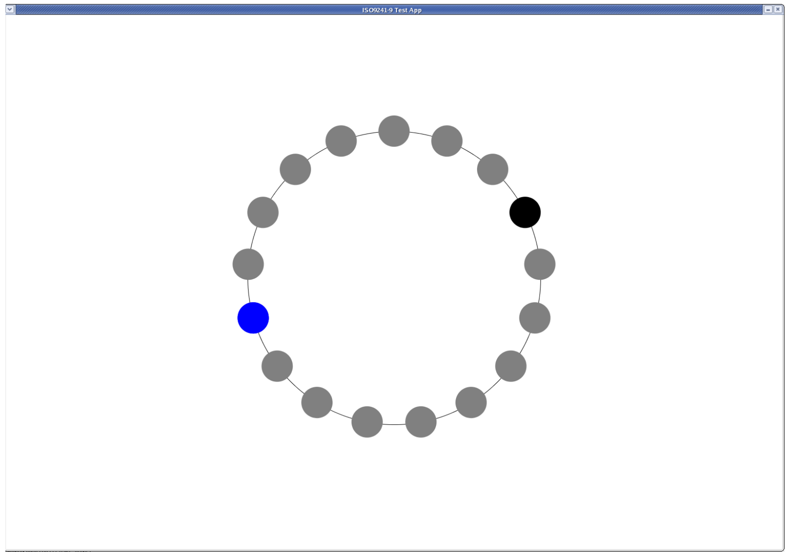


Figure 3.4: The abstract targets evaluation interface, the circle on the right is the current starting point, the circle on the left the target. The user must now select the target as quickly as possible.

with this name, and all log entries will include it also. Use of the interface is fairly simple. A circular array of greyed out targets is displayed, with one black target. The black target is the current starting position. To start a task, the user must hold the cursor over the starting position for 500ms, at which point the target on the opposite side of the circle changes to a bright blue colour. The user must then move the cursor to the opposite side of the circle and click the target as quickly as possible. The selection time is recorded as the time from when the cursor leaves the starting point to the time when the target is clicked. In addition to this, the time the cursor spent over the target before the click is also recorded. Following the click the target changes colour to indicate whether the selection was successful or not, green indicates a hit, while red indicates a miss. At this point an entry is added to the log, detailing the conditions of the selection. This includes the following: user name, selection time, mouse-over time, the index of difficulty of the task, the distance, the width, position and type of the target, the position of the click, and the block, array, and target numbers. Figure 3.4 shows the evaluation interface at the start of a targeting task, before the user has moved the cursor from the starting position. Figure 3.5 shows the evaluation interface when has just positioned the cursor over the target. It is only at this point that the user discovers the type of target they are selecting.

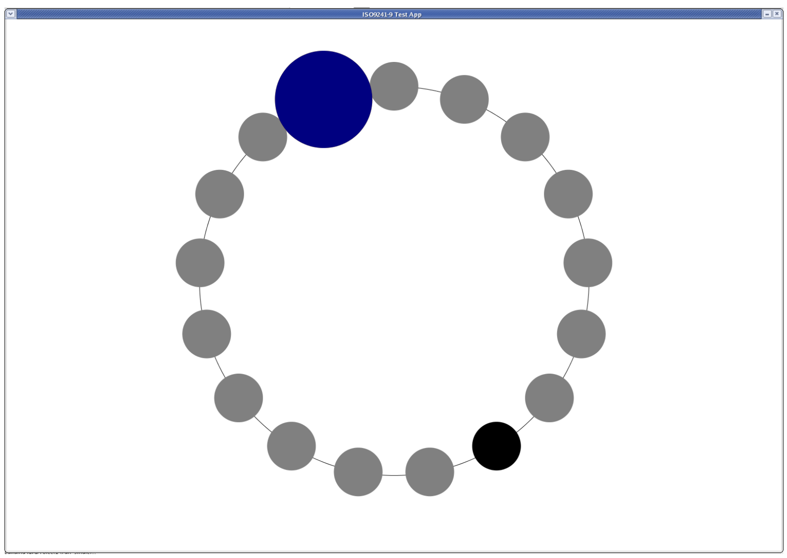


Figure 3.5: The abstract targets evaluation interface, the circle on the right is the current starting point, the circle on the left the target. The cursor has just entered the target, which has expanded and changed colour.

Chapter 4

Controlled Experiments

This chapter presents the results of two formal evaluations involving visually expanding targets. The first investigates the performance of visually expanding targets applied in an occluding fisheye menu system, comparing this system to traditional static menus, and rearranging fisheye menus. The second investigates whether visually expanding targets offer any selection advantage over non-expanding targets, and evaluates the results with respect to the well-established Fitts' Law.

The same participants took part in both experiments, and performed the second experiment immediately after the first.

4.1 Experiment One—Menu Evaluation

Visually expanding targets may offer selection advantages, however if they cannot be applied in real-world GUIs there is little point to them. This experiment is an ecological evaluation of visually expanding targets as applied to menu systems. As noted by Cockburn & Brewster (2005), menus are a good test platform, with densely-packed menu items exposing issues not always obvious in abstract targeting tasks. This experiment uses the menu testing interface described in Chapter 3 to evaluate and the performance of occluding fisheye menus with rearranging fisheye menus and traditional static-view JMenus.

4.1.1 Method

Participants completed two sets of tasks with three menu systems. In the first set of tasks, the evaluation interface displayed a cueing menu on the left side of the screen with the target menu item selected. Participants then selected the corresponding menu item from an identical menu positioned to the right of the cueing region. This cue is intended to simulate cases where the user is extremely familiar with the menu—to the extent that they know the spatial location of every item in the menu.

The second set of tasks displayed a word cue on the left side of the screen, participants then had to locate the word from the target menu on the right as quickly as possible. The contents of the menu changed for every task, as this cue was intended to simulate the use of menus unfamiliar to the user, as occurs when first using an application.

The time from posting the target menu to clicking the target item was recorded. In the case of an error, the background of the target menu region would turn red, and participants would then have to complete the task correctly. Error-rates are recorded, but do not negatively impact times, as the times logged are those of the successful selections, from posting the menu to selecting the target item. The length of time the cursor is over the target before selecting it is also recorded.

Although performed at the same time, the tasks cued with the word cue are distinct from those with the menu cue, and so are analysed independently.

Design

The experiment used two 3×3 repeated measures ANOVAs with factors "menu type" and "menu length", one for each cue type. The menu types were occluding fisheyes, rearranging fisheyes, and JMenus. The menu lengths were 10, 20 and 50 menu items. The dependent measures were selection time, mouse-over time, and error-rate.

Participants

The participants were postgraduate students and lecturers of Computer Science (ages 20–39 years, mean age 24 years, 15 male, 1 female), all from the University of Canterbury. All of the participants were advanced computer users, all were right-handed and all wore their best optical correction, if required. Each of the two experiments took approximately 30 minutes to complete, for a total of one hour's participation.

Participants were rewarded with heartfelt thanks, but nothing else of material worth.

4.1.2 Procedure

Each participant completed all of the menu cued tasks before completing the word cued tasks. With the menu cue, participants made 180 menu selections, in nine blocks of 20 selections (one block per menu length per menu type). The first ten selections in each block were practice tasks, and the data were discarded. Participants completed all of the blocks for a menu type before progressing to the next. The labels of the menu items differed randomly between participants and between each block, however the selection indexes were the same for each menu length. Menu types and selection tasks were presented to participants in a randomised order.

With the word cue, participants made 135 selections, in nine blocks of fifteen selection (one block per menu length per menu type). The first five selections in each block were practice tasks, and the data were discarded. Using the word cue, menus were randomly generated for every task, remaining in a constant order only until a successful selection click. Everything else was as stated for the menu cue.

Upon completing the experiment participants were asked to fill in a questionnaire to gain some subjective feedback on the menu types. Questions focussed on discovering which of the menu types was subjectively preferred, and which menu type participants thought made them most efficient for completing tasks with the two cue types. Additionally, participants were encouraged to write down any comments they had on the menus.

Apparatus

The experiment ran on Intel Pentium 4 2.8GHz computers with 1GB of RAM running the Fedora Core 3 Linux distribution. Graphics were supplied by GeForce FX5200 graphics cards driving 19inch Compaq monitors at 1600×1200 resolution, operating at 75Hz. Input was received through Labtec three-button mouse devices, with a one-to-one control-display gain setting. The evalutation interface was written using the Java 1.5 API, and ran full-screen.

Hypothesis

We believe that menu item selection times will be faster with occluding fisheye menus than with rearranging fisheye menus, with both types of cue (both menu cues and word cues) and across all menu length conditions (10, 20 and 50 menu items). We also believe that selection times using occluding fisheye menus will compare favourably to those using static menus, and will allow for faster selection times with the 50 menu item lengths. Participants will prefer using occluding fisheye menus over using both rearranging fisheye menus and static menus. Selection times for tasks involving the menu cue will be significantly faster than those for tasks with the word cue. Selection times for shorter menus will be less than those for longer menus, as indicated by Fitts' and Hick-Hyman Laws.

4.1.3 Results

There were 1440 logged tasks successfully completed for each cue type (16 participants using 3 menu types with 3 different menu lengths and 10 tasks for each condition). We will first present results for tasks using the menu cue, and follow this with results for tasks involving the word cue.

Menu Cue

Of the 1440 tasks, 67 were removed for being more than three standard deviations away from the mean time. Overall selection times were fast, the overall mean being 1071 milliseconds (s.d. 314).

There was a significant main effect for menu type, with JMenus having the fastest mean time of 936 milliseconds (s.d. 254), followed by occluding fisheye menus with 1051 milliseconds (s.d. 278), and rearranging fisheye menus being the slowest with 1226 milliseconds (s.d. 338) ($F=_{2,30}=74.8$, $p<.001$). There was an expected significant difference between menu length, as suggested by Fitts' law—the longer the menu, the further away the target is, the longer it takes ($F=_{2,30}=142.6$, $p<.001$)..

There was also a significant interaction between menu type and length. This is most likely caused by the deteriorating performance of the rearranging fisheye menus, as can be seen in Figure 4.1 the selection times for rearranging fisheye menus go up rapidly as menu length increases ($F=_{4,60}=3.7$, $p<.01$).

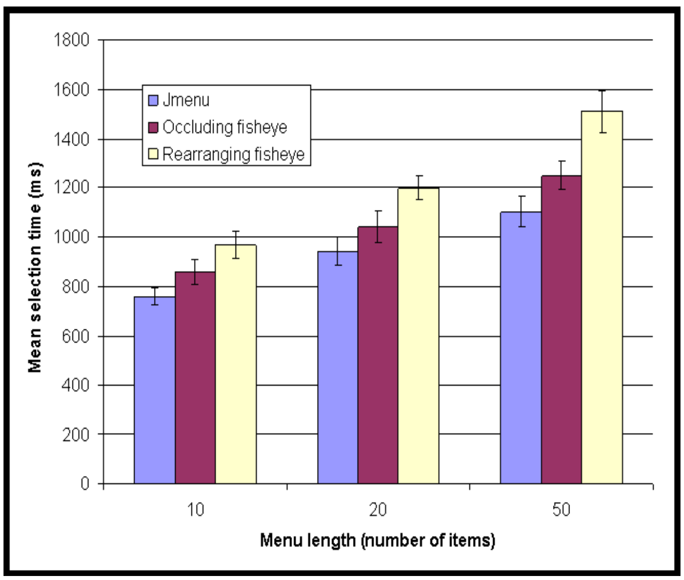


Figure 4.1: Mean selection times for the menu types. Error bars show the mean $\pm$ one standard error.

Word Cue

All 1440 results were log tranformed before analysis. There was significant difference between menu types, with JMenus having the fastest mean selection times at 3462 milliseconds (s.d. 1983), followed by occluding fisheye menus at 3350 milliseconds (s.d. 1967), with rearranging fisheye menus having the slowest selection times at 3462 milliseconds (s.d. 1964) ($F=_{2,30}=3.6$, $p<.05$). Again as expected there was a significant difference for menu lengths, with selections from shorter menus being faster, as suggested by the Fitts' and Hick-Hyman laws ($F=_{2,30}=740.0$, $p<.001$).

Subjective Measures

Participants answered three Likert-scale questions (1 for disagree, 5 for agree) after completing all tasks across all conditions. The mean responses to questions 1 and 2 "The menus were efficient for selecting items when using the menu/word cue" are shown in Table 4.1 The responses to question 1 gave a significant

Table 4.1: Means and standard deviations of participants' responses to questions 1 and 2: "The menus were efficient for selecting items when using the menu/word cue"

	Menu cue		Word cue	
Menu type	Mean	SD	Mean	SD
Static	4.75	0.58	3.56	1.09
Occluding fisheye	4.19	0.75	3.81	1.05
Rearranging fisheye	3.31	1.14	2.94	0.85

difference with participants finding static and occluding fisheye menus more efficient than rearranging fisheye menus when using the menu cue (Friedman Test, χ_r^2=13.34, df=2, p$<$.01). Question 2 found that participants rated occluding fisheye menus the most efficient and rearranging fisheye the least efficent when using the word cue, but this was not a significant difference (Friedman Test, χ_r^2=4.91, df=2, p=.086).

Table 4.2: Means and standard deviations of participants' responses to question 3: "I liked using the menus"

Menu type	Mean	SD
Static	4.19	0.83
Occluding fisheye	3.88	1.20
Rearranging fisheye	2.63	1.09

Question 3 asked participants to indicate whether they liked using the menus, and the recorded responses are shown in Table 4.2. This shows that the participants' feelings of like or dislike closely followed their evaluation of how efficient they were with the menus, resulting in a significant difference, with participants liking static and occluding fisheye menus, and disliking rearranging fisheye menus (Friedman Test, χ_r^2=8.66, df=2, p$<$.05).

Participants were asked to uniquely rank their preferences for the menus from 1–3 (1 most preferred, 3 least preferred), their menu rankings differed significantly (Friedman Test, χ_r^2=12.13, df=2, p$<$.01) with participants overall ranking static menus first, occluding fisheye menus second and rearranging fisheye menus last. Rearranging fisheye menus were ranked particularly lowly, with 81% of participants ranking it last. Static menus were ranked either first or second by all participants, and occluding fisheye menus were ranked either first or second by 81% of participants.

4.1.4 Discussion

Occluding fisheye menus offered significantly faster selection times than rearranging fisheye menus at all menu lengths for both tasks. Static JMenus outperformed them both, and were faster for both types of tasks. Subjectively, people felt they were more or equally efficient using static menus compared to occluding fisheye menus, and ranked static menus above both kinds of fisheye menus. Occluding fisheye menus were preferred over rearranging fisheye menus, and participants felt that for tasks involving searching for the word cue occluding menus were just as efficient as static menus.

Although slightly slower on average than static menus (12% slower on average for the menu cue, 3% slower on average for the word cue), occluding fisheye menus are a successful improvement over the original fisheye menu design, offering faster acquisition times and being subjectively preferred.

4.2 Experiment Two—Abstract Targets

The second experiment is an abstract target acquisition task, using four different combinations of feedback. Targets may be visually expanding or static, and they may be colour changing or remain the same colour.

The colour changing condition is included as a kind of visual feedback, as it is already well-established that feedback improves acquistion times (Cockburn & Brewster 2005), but we are interested in whether expansion offers improvements beyond that of simple visual feedback.

4.2.1 Method

The participants used the same apparatus as that in Experiment One, with the abstract targets testing interface described in Chapter 3.

The entire evaluation is conducted in the absence of anticipation of target type. Participants have no way of knowing what kind of feedback will be given prior to entering a target. This is motivated by a desire to discover a basic motor-level response that allows participants to select visually expandign targets more rapidly. To achieve this, participants complete four blocks of recorded tasks. Each block is made up of eight circular arrays of targets, with an array for each ID in each block. The target types are randomly permuted throughout each block in such a way that within each array there are an equal number of each target type, and upon completion of the four blocks, participants will have selected each target type once in each possible position of the circular array.

Design

The experiment was an $8\times2\times2\times4$ repeated measures ANOVA, with factors "index of difficulty", "visual expansion", "colour changing" and "block number". The eight levels of ID were 3.17, 3.18, 4.42, 4.48, 5.67, 5.71, 6.43 and 7.01, representing target distance and width combinations of (512, 64), (900, 112), (900, 44), (512, 24), (900, 18), (512, 10), (512, 6) and (1020, 8). Targets were either visually expanding or static, with half of each group also changing colour. The block number indicates how far through the experiment the task is, and is analysed for learning effects. The main dependent measure is selection time, but the mouse-over time is also analysed.

For the Fitts' law analysis, linear regressions between the movement time and index of difficulty were calculated for each of the target types.

4.2.2 Procedure

The participants were introduced to the evaluation interface and given a practice block before starting the evaluation. The practice block consisted of task using four circular arrays of targets with IDs similar to (but not the same as) those presented in the evaluation blocks. The IDs used spanned the range of those occurring in the evaluation, and the target types were selected randomly. The motivation for this was to avoid the participants developing expertise that favoured a particular target distance and width condition, while still allowing for them to gain familiarity with the task and interface.

Following the practice block, participants then completed four blocks of tasks, each block containing a circular array of targets for each ID. The order of the tasks within the blocks was randomised to avoid learning effects. Rach array of targets contained four of each of the target types, with one additional randomly selected practice "starter" target. The starter target allowed the participants to calibrate their movements for the given target distance and width, and the selection times for the starter target are discarded for the analysis.

Between each circular array of targets the participant's error-rate was shown, detailing error-rates for the last array, the last block, and over all. Participants were encouraged to maintain a 4% error-rate average, and were asked to speed up or slow down accordingly.

Upon completion of the evaluation, participants filled out a brief questionnaire.

Hypothesis

We believe that visually expanding targets will allow faster selection times than static targets, and that colour-changing targets will allow faster selection times than non colour-changing targets. Fitts' law will accurately model selection times across the different target distance and width conditions, for all types of targets. The amount of dwell-time spent over a target before clicking it will be less for visually expanding targets than for static targets, and less for colour-changing targets than for non colour-changing targets. We also believe that the difference in selection times between visually expanding and static targets will be greater than the difference in times between colour-changing and non colour-changing targets. We expect that visual expansion will help most with tasks with small targets and high indexes of difficulty. We also expect the inverse to be true of changes in colour, which will help more with larger targets due to the difficulty in perceiving the targets' change in colour at smaller sizes.

4.2.3 Results

There were 8192 task results logged, with 327 errors and 7865 successful selections (16 participants completing 4 blocks of 8 target arrays containing 16 targets). For the purpose of analysis, selection times over three seconds were discarded. We are looking at rapid acquisition tasks, and three seconds is more than three standard deviations above the mean acquisition time. This results in ten times from 7865 successful selections being discarded: three visually expanding, two visually expanding and colour changing, two static and three static and colour changing.

As expected, and predicted by Fitts' law, there was a significant main effect for ID, with selection times increasing with higher IDs ($F=_{1,15}=683.8$, $p<.001$). There was a significant main effect for visual expansion, with visually expanding targets being selected in 943 milliseconds (s.d. 329), while static targets mean selection time was 965 milliseconds (s.d. 356) ($F=_{1,15}=20.8$, $p<.001$) This confirms our hypothesis that visual expansion will help target acquisition. There was no significant difference in selection times between colour-changing and non colour-changing targets.

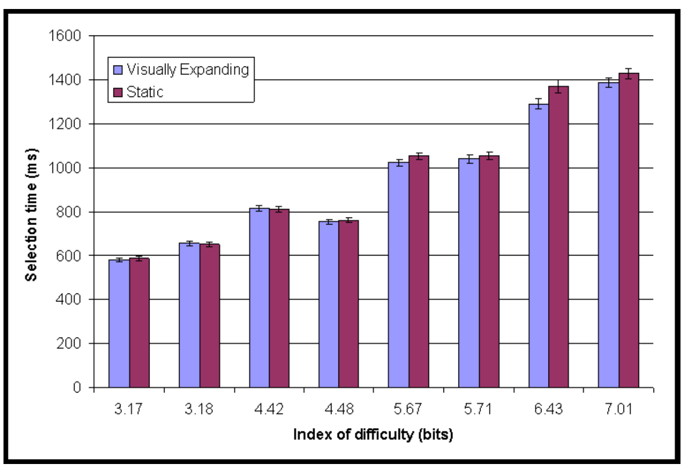


Figure 4.2: Mean selection times across IDs for visually expanding and static targets. Error bars show the mean $\pm$ one standard error.

There was a significant interaction between ID and visual expansion, with visual expansion improving acquisition times most for targets with high IDs, as shown in Figure 4.2 ($F=_{7,105}=4.7$, $p<.001$).

There was a significant difference for block number, with the first recorded block having an average selection time of 989 milliseconds (366 s.d.) and the final block having an average selection time of 934 ms (320 s.d.) ($F=_{3,45}=5.0$, $p<.01$). This effect fits well with the power law of practice (Anderson 1981), and makes sense, as participants complete more targeting tasks they become more adept at them. Closely related to this is the last significant interaction, between ID and block number. As the block number increases participants become faster at selecting targets with higher IDs ($F=_{21,315}=4.7$, $p<.01$). This is also a result of practice improving participants selection times, and as they are slower with smaller targets, there is more room for improvement.

Table 4.3: Linear regression equations, R^2 values, and p values for the target types. (MT = selection time in ms.)

Target type	Line of best fit	R^2	p
Static	$MT = 216.5 \times \text{ID} - 119.1$	0.96	>0.001
Static and colour changing	$MT = 216.2 \times \text{ID}$ - 117.8	0.95	>0.001
Visually expanding	$MT = 199.5 \times \text{ID}$ - 52.8	0.96	>0.001
Visually expanding and colour changing	$MT = 202.5 \times \text{ID}$ = 74.0	0.96	

Fitts' Law Models

Fitts' law accurately modelled selection times with all target types. Figure 4.3 shows the relationship between ID and mean selection times, with selection times rising with ID. Table 4.3 shows the linear equations delivered by the Fitts' analysis, all R^2 values are greater than 0.95, indicating a very good fit.

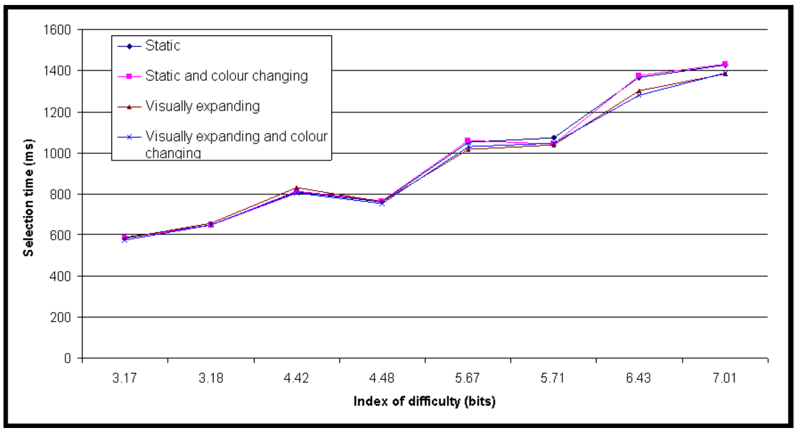


Figure 4.3: Mean selection times plotted against Index of Difficulty for the four target types.

Mouse Over Time

Using mouse-over time as the dependent variable results in another significant main effect for ID. As ID increases, so too does the length of time participants hover their mouse over the target before clicking it ($F_{7,105}$=115.2, $p<.001$). This is probably the result of participants taking longer to ensure accurate selection—because the target is smaller it is less easy to tell if the cursor is accurately positioned over it.

There is a significant difference for visual expansion, with visually expanding targets having a mean mouse-over time of 144 milliseconds (s.d. 75) and static targets having an average mouse-over time of 139 milliseconds (s.d. 72) ($F_{1,15}$=12.7, $p<.01$). This is surprising, and indicates that users took longer to select visually expanding targets than static targets once they were over the target.

There is a similar significant difference for colour changing, with colour changing targets having a mean mouse-over time of 143 milliseconds (s.d. 75) compared to non colour-changing with a mean mouse-over time of 140 milliseconds (s.d. 73) ($F_{1,15}$=7.1, $p<.05$). Again, this indicates that users took longer to select targets that were providing more feedback once they were over the target.

Misses

Error-rate is a crucial measure of performance in target acquisition tasks; an interface that speeds acquisitions but has a 50% error-rate is not particularly useful. An 8×2×2 ANOVA was used, with factors ID, visual expansion and colour changing, and the number of misses per condition as the dependent measure.

Across all conditions the error-rate was 3.99%, with 327 misses out of 8192 selections, which is remarkably close to the targeted rate of 4%. There was no significant difference in error-rate between expanding (173 misses) and static targets (154 misses), or colour-changing(160 misses) and non colour-changing(167 misses) targets. There was a significant difference in error-rate between IDs, as shown in Figure 4.4. This is explained by the much smaller target sizes of the high ID conditions, which naturally leads to a greater number of misses. There were no other significant interactions.

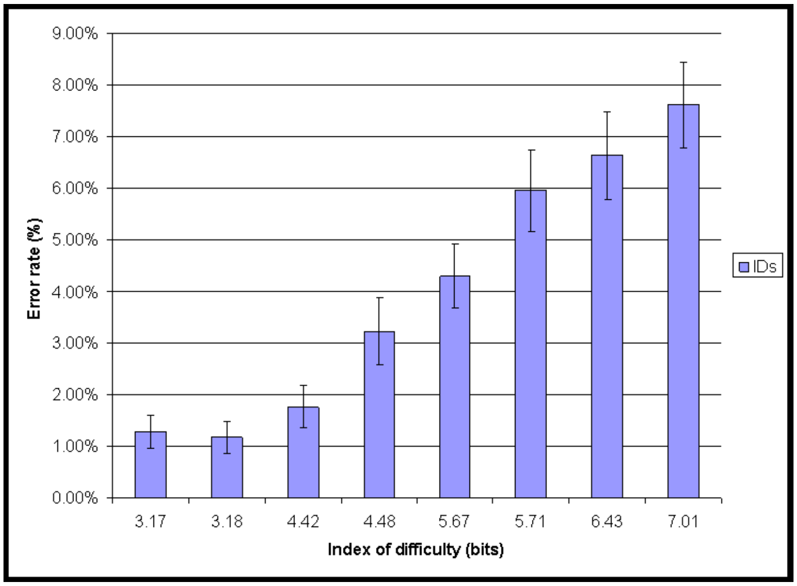


Figure 4.4: Total error rates across IDs. Error bars show the mean $\pm$ one standard error.

Subjective Measures

Participants answered two Likert-scale questions (1 for disagree, 5 for agree) after completing all tasks across all conditions. The mean responses to question 1 "I was efficient selecting the targets" are shown in Table 4.4 The responses to question 1 gave a significant difference with participants finding themselves more efficient at selecting visually expanding targets than static targets, and more efficient with coloured targets than uncoloured targets (Friedman Test, χ^2_r=25.56, df=2, p<.001).

Table 4.4: Means and standard deviations of participants' ratings of efficiency with the various target types

Target type	Mean	SD
Visually expanding	4.06	0.68
Visually expanding and colour changing	4.44	0.63
Static	2.63	0.89
Static and colour changing	3.38	0.89

Question 2 asked participants to indicate whether they liked selecting the various types of targets, and the recorded responses are shown in Table 4.5. As with the first experiment, participants' feelings of like or dislike correspond closely to their evaluations of efficiency, and resulted in a significant difference, with participants liking visually expanding targets more than static targets, and again rating coloured targets higher than uncoloured (Friedman Test, χ^2_r=15.02, df=2, p<.01).

Participants were asked to uniquely rank their preferences for the target types from 1–4 (1 most preferred, 4 least preferred), and ranked visually expanding and colour changing targets first, visually expanding targets second, static and colour changing targets third and static targets last. These rankings differed

Table 4.5: Means and standard deviations of participants' ratings of how much they liked selecting the various target types

Target type	Mean	SD
Visually expanding	3.94	1.18
Visually expanding and color changing	4.13	1.15
Static	2.44	0.73
Static and color changing	3.31	1.01

significantly (Friedman Test, χ_r^2=16.50, df=2, p$<$.001), with 75% of participants ranking static targets as their least preferred, and 75% of participants ranking one of the visually expanding target types as their first choice.

Discussion

As hypothesised, visual expansion significantly improved target acquisition times. Strangely, the mouse-over times for expanding targets is greater than that for static targets, showing that users spend more time with the cursor positioned over the target when it expands than if it remains a constant size. Despite this, selection times are faster with the expanding targets, even though the expansion only occurs once the cursor has entered the target! A possible explanation of this could be that participants rapidly move the cursor towards target, to the point where they cross over it, and upon seeing the visual feedback of the expansion (or alternatively the colour change, for both feedback techniques showed the same effect) realise they are close to the target and rapidly hone in on it. Once on the target however, they spend a fraction of a second to verify the feedback before clicking. When approaching a static target no feedback is given as the cursor crosses over, resulting in a slower acquisition time. Once the participant believes he is over the target however there is no feedback to wait for, and no reason not to click immediately.

Chapter 5

Discussion and Further Work

We have implemented occluding fisheye menus and shown that they perform better than the original fisheye menus, and are only slightly slower than standard static menus.

The evaluation of visually expanding targets showed an interesting effect, whereby expanding targets helped target acquisition, yet seemingly not at the point in the selection process expected. A tentative explanation for this is that the visual feedback of the expansion allows users to more quickly hone in on the target, however we have insufficient data to confirm this. Despite concerns of a fatigue effect, the abstract targeting task showed decreasing selection times as the evaluation progressed, as participants became more adept at selecting smaller targets.

Participants seemed to recognise the conditions where visual feedback helped, even with the relatively small difference it made in selection times, with many making comments such as "Visually expanding helps a lot with the really tiny targets. Colour changing only helps with the medium to larget targets, as the mouse cursor covers a lot of the small targets' surface area."

A number of participants commented on the length of the experiment, and expressed their dislike of the tasks. Two participants independently gave comments saying "I hate you. I want you [to] die." upon completion of the evaluation.

5.1 Further Work

5.1.1 Abstract Targets – Experimental Design

The experiment we conducted with abstract targets investigated the use of visually expanding targets to aid acquisition *in the absence of expectation*. This meant participants had no way of predicting whether the next target would undergo expansion or not, and indeed, had no way of knowing which of the four types of feedback would be presented. Effectively this meant we were looking at a very low level motor-response to the feedback, as people cannot anticipate the response prior to positioning the mouse over the target. The results show that visual expansion can in fact help with target acquisition, particularly when acquiring small targets. Given the length of the experiment already, and participants' reluctance to undertake any further tasks,[1] we were unable to evaluate the impact of visual expansion when it is expected.

Further investigation is thus required to discover what benefits there are to visual expansion when people can anticipate that it is going to occur. This could be as simple as repeating the evaluation we conducted without the randomised permutation of feedback conditions. A possible concern is the order of presentation of the feedback types, as even despite the simple nature of the task, and the practice block at the start of the evaluation, a learning effect was still observed. We anticipate that the benefits of visually expanding targets will be even more evident when they are presented in blocks and participants can predict their occurrence.

[1] One of the participants expressed this clearly with the following comment: "No, I'm not doing a longer evaluation."

5.1.2 Fisheye Menus – Cueing Systems

The evaluation of fisheye menus used two different cueing systems, designed to reflect two different kinds of tasks performed with menus. The task menu cue was meant to simulate situations where a user has significant familiarity with a menu through frequent use, and thus has highly developed spatial memory of the position of various menu items. The word cue simulated situations where a user knows the name of the menu item they are looking for, but is unfamiliar with the menu layout. This situation can occur when an application is used infrequently, or is being used for the first time. An alternative cue would be to highlight the background of the target menu item, within the active menu itself. This could simulate situations where the user has visually acquired the desired menu item prior to selecting it, and would provide more accurate menu performance data for those people who do this visual scan prior to moving the mouse. Any future evaluations should include all three cueing types to assess the performance of the menus.

5.1.3 Fisheye Menus – Cascading Menus

The evaluation of fisheye menus dealt with menus using only one level of menu items. Frequently menus contain cascading submenus, creating a hierarchy of different menu items. Were fisheye menus to become widely applied, they would need to support this hierarchical structure, and so an evaluation of cascading fisheye menus is necessary to ascertain their performance and viability as a static menu replacement.

5.1.4 Fisheye Menus – Long Menus

This paper has been centred on the use of visually expanding targets, rather than purely on the applications of fisheye menus. In the original fisheye menu paper (Bederson 2000) much of the focus was on evaluating fisheye menus as a means of supporting longer menus than can be obtained using purely static menus. This led to comparisons between fisheye menus, hierarchical menus, arrowbar menus (whereby the user can scroll up and down the menu by pressing arrow buttons at the top and bottom of the menu) and scrollbar menus (whereby the user can scroll up and down the menu using a scrollbar). In Bederson's evaluation, fisheye menus were only out-performed by hierarchical menus. This paper has shown that occluding fisheye menus offer significant gains over the rearranging fisheye menus proposed by Bederson, at least for menus with 50 items or less. Given this promising outcome, further investigation of the use of occluding fisheye menus as a paradigm for long menus (holding from 50 to 200 menu items) is warranted.

5.1.5 Fisheye Menus – the Vision Impaired

Although occluding fisheye menus appeared to offer no improvement on static menus, performing 3% to 12% slower on average than static menus with the menu cue and word cue respectively, there exists the possiblity that they will provide more of a benefit to people with a significant vision impairment. Currently, the only way to make menu items in static menus appear larger[2] is to make the font of all of the items in the menu larger. This has an immediate limiting effect on the potential length of menus by reducing the number of items that can possibly fit on screen at a time. Occluding fisheye menus offer the best of both worlds, allowing for the same number of items as available in a smaller font-sized static menu, while also providing an enlarged, more readable selection area. An additional study investigating the use of fisheye menus by the vision impaired should be conducted.

5.1.6 Fisheye Menus – Longitudinal Study

A key concern with an HCI evaluation of this type is that the participants have often been exposed to some of the interfaces for disproportionately longer than some of the others. This is particularly relevant when evaluating alternatives to current, widely-used interfaces. A common technique for reducing this effect is to provide practice tasks for participants to complete to allow them to gain some proficiency with the various interfaces. As shown (Ohlsson 1996), practicing a task allows for improvements in the performance of that task—with both time taken and error rates dropping off along negatively accelerated curves. This effect is

[2] Without the use of some other magnifying widget/device.

commonly known as the power law of practice (Anderson 1981) and can be seen in action with almost any task performed by humans. Given this behaviour, participants can be expected to quite rapidly attain a skill level where comparisons can be drawn, and this puts the goal of comparing interfaces within reach.

Thus we are able to compare fisheye menus with static menus after only a short practice period. However, although the results show occluding fisheye menus were slower than static menus, it is still the case that participants have had significant experience with static menus and only minimal experience with fisheye menus. A longitudinal study with participants exposed to fisheye menus for a longer period of time could see them developing new strategies that allow for faster menu usage. This kind of improvement is supported by cognitive theories such as ACT-R (Anderson & Lebiere 1998), which says that a series of actions can be streamlined into a single action combining them, and thus more rapidly executed. Essentially a longitudinal study allows for a much extended practice interval and progression along the practice curve to where participants can be considered experts at both menu types. Beyond the possible improvements in speed of use, the participants' subjective preferences would be less likely to be affected by the novelty factor of fisheye menus, and more likely to reflect their actual usefulness.

Chapter 6

Conclusions

We have conducted an evaluation of visually expanding targets in an abstract targeting task, comparing expanding targets with a colour-changing visual feedback. We have presented results showing that visual expansion can result in faster target acquisition times, particularly for tasks involving small targets.

We have implemented an alternative fisheye menu system using in place expansion and conducted an ecological evaluation of visually expanding targets, comparing the occluding fisheye menu with standard menus and the original fisheye menu system. We have presented results showing that the occluding fisheye system allows for faster target acquisitions than the original fisheye system both for tasks involving spatial memory, and for tasks searching unfamiliar menus. The occluding fisheye menu is slightly slower on average than a standard JMenu, however this could be due to participants' extensive exposure to static menus.

Visually expanding targets with no change in motor-space do not appear to negatively affect target acquisition, and could be used (with caution) to provide more interesting GUIs.

Occluding fisheye menus show promise, allowing similar selection times to standard menus, and could be used as an optional 'cool' effect, not unlike the MacOS X dock.

6.1 ACKNOWLEDGMENTS

A huge thank you to all those long–suffering people who took part in the experiment—your clicks made this report possible.

Bibliography

Accot, J. & Zhai, S. (1997), Beyond fitts' law: models for trajectory-based hci tasks, *in* 'CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 295–302.

Ahlstrom, D. (2005), Modeling and improving selection in cascading pull-down menus using fitts' law, the steering law and force fields, *in* 'CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 61–70.

Anderson, J. R. (1981), *Cognitive Skills and Their Acquisition*, Lawrence Erlbaum Associates, Inc.

Anderson, J. R. & Lebiere, C. (1998), *The Atomic Components of Thought*, Lawrence Erlbaum Associates, Inc.

Bederson, B. (2000), Fisheye Menus, *in* 'Proceedings of the 2000 ACM Conference on User Interface Software and Technology, San Diego, California.', pp. 217–225. http://www.cs.umd.edu/hcil/fisheyemenu/.

Blanch, R., Guiard, Y. & Beaudouin-Lafon, M. (2004*a*), Semantic pointing: improving target acquisition with control-display ratio adaptation, *in* 'Proceedings of CHI'2004 Conference on Human Factors in Computing Systems Vienna, Austria', ACM Press, pp. 519–526.

Blanch, R., Guiard, Y. & Beaudouin-Lafon, M. (2004*b*), Semantic pointing: improving target acquisition with control-display ratio adaptation, *in* 'CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 519–526.

Byrne, M. D., Anderson, J. R., Douglass, S. & Matessa, M. (1999), Eye tracking the visual search of click-down menus, *in* 'CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 402–409.

Callahan, J., Hopkins, D., Weiser, M. & Shneiderman, B. (1988), An empirical comparison of pie vs. linear menus, *in* 'CHI '88: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 95–100.

Card, S. K. (1982), User perceptual mechanisms in the search of computer command menus, *in* 'Proceedings of the 1982 conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 190–196.

Cockburn, A. & Brewster, S. (2005), 'Multimodal feedback for the acquisition of small targets', *Ergonomics* **48**(9), 1129–1150.

Cockburn, A. & Firth, A. (2003), Improving the Acquisition of Small Targets, *in* P. Palanque, P. Johnson & E. O'Neill, eds, 'People and Computers XVII (Proceedings of the 2003 British Computer Society Conference on Human-Computer Interaction.)', Springer-Verlag, pp. 181–196. `www.cosc.canterbury.ac.nz/~andy/papers/smallTargets.pdf`

Cockburn, A., Savage, J. & Wallace, A. (2005), Tuning and testing scrolling interfaces that automatically zoom, *in* 'CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 71–80.

Findlater, L. & McGrenere, J. (2004), A comparison of static, adaptive, and adaptable menus, *in* 'CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 89–96.

Fitts, P. (1954), 'The information capacity of the human motor system in controlling the amplitude of movement.', *Journal of Experimental Psychology* **47**, 381–391.

Furnas, G. (1986), Generalized Fisheye Views, *in* 'Proceedings of the CHI'86 Conference on Human Factors in Computing Systems III', Amsterdam; North Holland/ACM, pp. 16–23.

Gann, K. & Hoffmann, E. (1988), 'Geometrical conditions for ballistic and visually controlled movements', *Ergonomics* **31**(5), 829–839.

Grossman, T. & Balakrishnan, R. (2005), The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area, *in* 'CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 281–290.

Guiard, Y., Blanch, R. & Beaudouin-Lafon, M. (2004), Object pointing: a complement to bitmap pointing in guis, *in* 'GI '04: Proceedings of the 2004 conference on Graphics interface', Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, pp. 9–16.

Gutwin, C. (2002), Improving Focus Targeting in Interactive Fisheye Views, *in* 'Proceedings of CHI'2002 Conference on Human Factors in Computing Systems. CHI Letters 4(1). Minneapolis, Minnesota, 20–25 April', pp. 267–274.

Hick, W. E. (1952), 'On the rate of gain of information', *Quarterly Journal of Experimental Psychology* **4**, 11–36.

Hyman, R. (1953), 'Stimulus information as a determinant of reaction time', *Journal of Experimental Psychology* **45**, 188–196.

Kobayashi, M. & Igarashi, T. (2003), Considering the direction of cursor movement for efficient traversal of cascading menus, *in* 'UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology', ACM Press, New York, NY, USA, pp. 91–94.

Kurtenbach, G. & Buxton, W. (1994), User learning and performance with marking menus, *in* 'CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 258–264.

Landauer, T. K. & Nachbar, D. W. (1985), Selection from alphabetic and numeric menu trees using a touch screen: breadth, depth, and width, *in* 'CHI '85: Proceedings of the SIGCHI conference on Human factors in computing systems', ACM Press, New York, NY, USA, pp. 73–78.

MacKenzie, I. (1992), Movement time prediction in human-computer interfaces, *in* 'Proceedings of Graphics Interface '92 Toronto', Canadian Information Processing Society, pp. 140–150.

McGuffin, M. & Balakrishnan, R. (2002), Acquisition of Expanding Targets, *in* 'Proceedings of CHI'2002 Conference on Human Factors in Computing Systems. CHI Letters 4(1). Minneapolis, Minnesota, 20–25 April', pp. 57–64.

Mitchell, J. & Shneiderman, B. (1989), 'Dynamic versus static menus: an exploratory comparison', *SIGCHI Bull.* **20**(4), 33–37.

Moyle, M. & Cockburn, A. (2003), The design and evaluation of a flick gesture for 'back' and 'forward' in web browsers, *in* 'CRPITS '03: Proceedings of the Fourth Australian user interface conference on User interfaces 2003', Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 39–46.

Nilsen, E. & Evans, J. (1999), Exploring the divide between two unified theories of cognition: modeling visual attention in menu selection, *in* ‘CHI ’99: CHI ’99 extended abstracts on Human factors in computing systems’, ACM Press, New York, NY, USA, pp. 288–289.

Ohlsson, S. (1996), ‘Learning from performance errors’, *Psychological Review* **103**, 241–262.

Sears, A. & Shneiderman, B. (1994), ‘Split menus: effectively using selection frequency to organize menus’, *ACM Trans. Comput.-Hum. Interact.* **1**(1), 27–51.

Somberg, B. L. (1987), A comparison of rule-based and positionally constant arrangements of computer menu items, *in* ‘CHI ’87: Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface’, ACM Press, New York, NY, USA, pp. 255–260.

Soukoreff, R. & MacKenzie, I. (2004), ‘Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts’ law research in HCI’, *International Journal of Human-Computer Studies* **61**(6), 751–789.

Zhai, S., Conversy, S., Beaudouin-Lafon, M. & Guiard, Y. (2003), Human On-line Response to Target Expansion, *in* ‘Proceedings of CHI’2003 Conference on Human Factors in Computing Systems Fort Lauderdale, Florida’, pp. 177–184.